**The SIZER Utility Program**

*Version 1.03*

Dave Surman

[surman@us.ibm.com](mailto:surman@us.ibm.com)

Bill Neiman

[neiman@us.ibm.com](mailto:neiman@us.ibm.com)

IBM Poughkeepsie

09/26/2025

**Overview**

The SIZER utility is an unsupported, as-is utility program intended to help the installation to plan for and understand coupling facility (CF) structure size changes that may be required when migrating CFs from one CFLEVEL to another.

When run, the SIZER utility will inspect all of the CF structure instances that are currently allocated, in all the CFs that are accessible from the system where the utility is run, analyze and report on their detailed CF structure attributes (in CF architecture terms), and then determine and report on the structure size for the structure in several different ways:

* The CFRM *policy* size-related definitions and/or defaults for the structure
* The current, *actual* allocated structure sizing information
* The *calculated* sizing information for a structure having identical attributes to the allocated structure instance, if it were to be allocated in each of the other accessible CFs in the installation that are accessible to the system where the utility is run. CFs that are not connected to the system, or which are not managed (e.g. not defined in the current active CFRM policy) will not be processed.   
    
  In effect, this is a “what-if” calculation to determine what the sizing requirements would be for this structure if it were to be allocated in each of the other CFs in the configuration, at whatever CFLEVEL those CFs may be.

The minimum CFLEVEL supported by the SIZER utility is CFLEVEL 8. This level of coupling facility is the lowest level that supports the CF commands used in performing the “what-if” structure size calculations.

The output of the SIZER utility may be useful to you in determining how to make CFRM policy structure size definition changes (SIZE, INITSIZE, and MINSIZE parameters) to accommodate CF structure size growth between CFLEVELs. (See the CFSizer website at <https://www.ibm.com/support/pages/cfsizer> for additional general information about CF structure sizing and requirements).

**Note that in order for the SIZER utility to be useful for this re-sizing purpose, you must already have provided a CF at the new “uplevel” CFLEVEL in your installation, and it must be configured and active in your sysplex, with connectivity to the system on which the SIZER utility is being run.**

Also note that if more than one active instance of a particular structure is allocated at the time that the SIZER utility is run (e.g. because the structure is duplexed, or is in the process of being rebuilt), each of the structure instances will be reported on separately.

It may be useful to know that the SIZER utility makes use of several general-use Sysplex services to retrieve the structure information it uses and provides. These services include:

* IXLMG and the IXLYAMDA output area mapping
* IXLCSP and the IXLYCSPA output area mapping
* IXCQUERY and the IXCYQUAA output area mapping.

Wherever possible, the source of the information described (in terms of the above services and output areas) is provided below.

**Usage Scenario**

The SIZER utility is primarily intended as a migration aid when migrating CFs to a new CFLEVEL. Note the following points:

* As noted above, **you must have a CF at the new CFLEVEL installed in your configuration and accessible from the system where the utility is being run.**
* **SIZER is useful when you are satisfied that your structures are sized appropriately for your current workload at your current CFLEVEL, and you only want to ensure that the existing capacity is preserved on migration to the new CFLEVEL.**
* The utility itself has no dependency on CFLEVEL and does not require update when a new CFLEVEL is introduced (unless the new CFLEVEL introduces some specific architectural change that affects structure sizing, such as introducing a new kind of structure object). In general, you do not need to download a new version of SIZER when you migrate to a new CFLEVEL.
* The utility has no dependency on z/OS level, and will run on any supported z/OS release.

To use SIZER appropriately, a typical sequence would be:

1. Drain the outgoing CF.  
     
   Move all structures out of the CF by rebuilding, stopping duplexing rebuilds, or otherwise deallocating structures in accordance with the procedures established by the associated application. You may find it helpful to place the outgoing CF in maintenance mode (SETXCF START,MAINTMODE,CFNAME=cfname) to ensure that no structures are reallocated in the CF during this process. It may be necessary to update structure preference lists in the CFRM policy to establish the desired structure distribution. With the CF in maintenance mode and / or preference lists updated, the REALLOCATE process (SETXCF START,REALLOCATE) will be the most efficient way to redistribute the structures.  
     
   This step assumes that you have other online CFs into which you can move the structures. If you have only one CF, and are upgrading that CF via a push-pull process, you will have to leave the structures in the CF at this point and drain it after you make the new CF available and run the SIZER utility in step 3.
2. Bring the uplevel CF into use, but do not populate it.  
     
   Update the CFRM policy to remove the outgoing CF and define the incoming CF, and then start the updated policy (SETXCF START,POLICY,TYPE=CFRM, POLNAME=polname). Do not change policy sizes or move structures into the new CF at this point.
3. With the structures in the remaining downlevel CF(s) and the uplevel CF still empty, run the SIZER utility to determine appropriate structure sizes for the new CFLEVEL.  
     
   At this point, you must have both old and new CFs connected and accessible to the system from which you are running the utility, and both CFs must be defined in the CFRM policy. (Even during a push-pull upgrade, there must be a point in the procedure where both CFs are connected and accessible.)
4. Update the CFRM policy to reflect the new sizes only for those structures that will be moved into the upgraded CF, and start the updated policy.   
     
   If you modified structure preference lists in the CFRM policy in earlier steps to enforce a desired structure distribution, you can restore them as you update the policy size values. The policy will remain pending until the affected structures are rebuilt into the new CF. (Again, the REALLOCATE command is the most efficient way to accomplish this.) The rebuild has the dual effect of activating the new sizes and moving the structures to their desired location.

Note that there is only one policy start across steps 3 and 4. Do not start a CFRM policy with the newly-calculated sizes until you are actually ready to move the structures, or you will find that the structure sizes increase to more than the calculated values as discussed in the following example.

Example of incorrect procedure – don’t do it this way!

1. With the structures in the remaining downlevel CF(s) and the uplevel CF still empty, run the SIZER utility to determine appropriate structure sizes for the new CFLEVEL. (Same as above – OK so far.)
2. Update the CFRM policy to reflect the new sizes in preparation for moving the structures into the upgraded CF, and start the updated policy.   
     
   This seems like a prudent procedure, but it’s incorrect. Here’s why: When you start the updated policy, it will remain pending until the affected structures are rebuilt. So you rebuild the structures in place, and now the structures reside in the downlevel CF(s) but with sizes calculated for the uplevel CF. The SIZER utility has determined the sizes necessary to preserve the existing numbers of structure objects (list headers, list entries, data elements, cache directory entries, locks, etc.) in the uplevel CF. When you reallocate those structures in the downlevel CF at the new sizes, those structures now contain more structure objects than they did before. They’re bigger than they need to be at the lower CFLEVEL.
3. Update the preference lists in the CFRM policy as necessary, and rebuild to move structures into the uplevel CF.  
     
   Since there is no policy size change pending (because you diligently but inappropriately prepared the way by increasing sizes in step 4), this rebuild is done “by counts” rather than “by size”. It ignores the CFRM policy sizes, and instead performs the rebuild so as to preserve the existing counts of structure objects. However, the number of objects has been increased beyond what you wanted because of the policy update and rebuild in step 4, and preserving the larger object counts results in structure sizes larger than intended.

Thus, the correct procedure is to perform a single CFRM policy update that both updates sizes and redistributes the structures in one step.

**Installation and Execution**

Download the SIZER utility package from the alternate sizing techniques page of the CFSizer website, at <http://www.ibm.com/systems/support/z/cfsizer/altsize.html>. The package contains a single-CSECT object file, sample link-edit JCL, sample execution JCL, and this documentation, compressed into a single file. After expanding the downloaded file, upload the object file and the two JCL jobs to the host as FB80 binary files. Edit the link-edit JCL, customize the JOB statement as desired, update the OBJ DD to point to the uploaded object file, and update the LNK DD to specify the target library, which must be APF-authorized. Run the link-edit job to create the executable load module.

Before running the SIZER utility, edit the execution JCL, customize the JOB statement as desired (or optionally remove it if you intend to run the utility as a started task), and customize the SIZEROUT DD to specify the destination for the output. Output can be collected in a pre-allocated or newly-allocated VB259 data set, or directed to SYSOUT.

You can run the utility as a batch job or as a started task. The system from which you run the utility:

* Must have connectivity to all CFs that are to be included in the analysis.
* Must be at a release or service level that supports CF use of storage-class memory (SCM) if any allocated structure is SCM-capable. If the utility is run on a down-level system, SCM-related values will not be presented.
* Must be at a release or service level that supports asynchronous system-managed duplexing if any allocated structure is currently duplexed using that protocol or if the CFRM policy permits it to be. If the utility is run on a down-level system, calculated structure sizes will not account for the additional controls required to support asynchronous duplexing.

**Storage-Class Memory**

Beginning with zEC12 GA2 (CFLEVEL 19) and ending with z16 (CFLEVEL 25), it is possible to configure a CF LPAR with SCM (flash memory) to provide an overflow capacity for some structures.

**NOTE:** Coupling facilities defined on the z17 processor or later (CFLevel 26 and above) no longer support storage-class memory. Refer to application-specific documentation for alternative means of providing structure overflow capacity.

SCM is available only for list structures having specific attributes, such as MQSeries application structures. The CFRM policy SCMMAXSIZE and SCMALGORITHM keywords designate a structure as eligible to use SCM. In the remainder of this document, the term “SCM-capable structure” means an SCM-eligible structure that is allocated in a CF which is at a CFLEVEL between 19 and 25 inclusive and configured with SCM.

When a structure is SCM-capable, the CF can migrate structure objects from CF real storage into SCM to avoid structure full conditions. Use of SCM has important implications for sizing structures:

* The maximum amount of SCM a structure is permitted to use, specified by the SCMMAXSIZE keyword, directly affects the amount of CF real storage required by the structure. Internal CF control structures must increase in size to accommodate the use of SCM, and the more SCM is authorized to the structure, the larger the structure itself must be.
* Structures that use SCM are required to provide sufficient capacity for a specific minimum number of structure objects to ensure effective use of SCM. If a structure were to have an insufficient number of objects, the CF’s migration and pre-fetch algorithms for moving objects between real storage and flash would not work efficiently. The result of this requirement is to impose an effective minimum size on any SCM-capable structure, which in practice, at CFLEVEL 19, is approximately 300M.
* SCM is not permanently associated with a structure. When needed, it is obtained from the pool of configured SCM, and when no longer needed, it is released. It is therefore possible to create a CFRM policy that “overcommits” the SCM configured to a given CF by authorizing multiple structures to use a total amount of SCM that exceeds the configured amount. The SIZER utility always assumes that the full amount of SCM authorized by the CFRM policy or configured to the CF is available to the structure.
* Use of SCM requires additional CF real storage called “augmented space”, which is used to manage and locate structure objects in SCM. An SCM-capable structure always has a small amount of fixed augmented space associated with it. Additional augmented space is obtained and released as needed. Neither the fixed nor the dynamic augmented space is included in the structure size, but it nevertheless imposes a potentially significant requirement for CF real storage which must be accounted for when planning CF capacity.
* The amount of overflow capacity that is provided by a given amount of SCM and the amount of augmented space necessary to support that capacity can only be estimated. The actual values depend on the usage pattern of the structure.

**Asynchronous Duplexing**

Beginning with z13 GA2 (CFLEVEL 21), certain structures can be duplexed using an asynchronous protocol in which the primary CF is responsible for forwarding updates to the secondary CF. Structures that support the asynchronous duplexing protocol require additional internal controls that increase the size of the structure. As of the date of this document, only lock structures can be asynchronously duplexed.

**Interpreting the SIZER Utility’s Output**

The SIZER utility generates a report with information about every currently-allocated structure instance, in every CF in your installation that is accessible to the system where the utility is run. A representative sample report would look something like the following:

SIZER: Processing started - VERSION 1.02 – 09/30/16

----------------------------------------------------------------------------------------

Structure DSNDBSG\_LOCK1

Allocated in CF cfname [(rebuild old|new structure)]

Lock structure

Immediate RTC.. NO Wait on RTC.... No

RData entries.. 84340 NumUsers....... 7

Lock entries... 16777216 List count..... 113

Duplex cntls... 4096 DUPLEX modes... SYNCONLY,ASYNC

CFNAME CFLVL CFSRV MIN/MASS STRSIZE MAXSIZE

-------- ----- ----- --------- --------- ---------

POLICY - - 48000K 64000K 140000K

CURRENT - - 42M 63M 137M

CF4 18.00 04.04 CF not ASYNCDUPLEX-capable

Without duplex controls: 63M 137M

CF2 17.00 02.13 CF not ASYNCDUPLEX-capable

Without duplex controls: 62976K 137M

CF1 19.00 00.15 CF not ASYNCDUPLEX-capable

Without duplex controls: 63M 137M

CF3 21.00 02.16 N/A 68M 137M

----------------------------------------------------------------------------------------

Structure CQS\_FF\_LOGSTR

Allocated in CF cfname [(rebuild old|new structure)]

List structure

Key/Name....... Entry keys Adjunct........ Yes

List accounting Elements PLEIDs......... Yes

Immediate RTC.. No Wait on RTC.... No

ElemChar....... 1 MaxElemNum..... 128

List count..... 6 Lock Entries... 0

Entries........ 13158222 Elements....... 39473403

Pend ent ratio. 1 Pend elem ratio 3

Pend EMCSTG.... 0 SLND........... 0

CFNAME CFLVL CFSRV MIN/MASS STRSIZE MAXSIZE

-------- ----- ----- --------- --------- ---------

POLICY - - 0M 0M 25G

CURRENT - - 440M 24904M 25G

CF4 18.00 04.04 N/A 24486M 25G

CF2 17.00 02.13 N/A 24088M 25G

CF1 19.00 00.15 N/A 24786M 25G

CF3 21.00 02.16 N/A 24786M 25G

----------------------------------------------------------------------------------------

Structure DSNDBWG\_GBP30

Allocated in CF cfname [(rebuild old|new structure)]

Cache structure

Name cls mask.. FC00 Adjunct........ No

UDF order...... Yes Immediate RTC.. No

ElemChar....... 4 MaxElemNum..... 1

COClasses...... 1024 StgClasses..... 1

Entries........ 90114 Elements....... 22526

Pend ent ratio. 4 Pend elem ratio 1

Free dir ent... 57702 Free data elem. 1463

GCUDRI......... 11779

CFNAME CFLVL CFSRV MIN/MASS STRSIZE MAXSIZE

-------- ----- ----- --------- --------- ---------

POLICY - - 93748K 125000K 250000K

CURRENT - - 6M 123M 245M

CF4 18.00 04.04 N/A 123M 245M

CF2 17.00 02.13 N/A 125440K 245M

CF1 19.00 00.15 N/A 123M 245M

CF3 21.00 02.16 N/A 123M 245M

----------------------------------------------------------------------------------------

Structure MQGPBOOK

Allocated in CF CF3

List structure

Key/Name....... Secondary keys Adjunct........ Yes

List accounting Entries PLEIDs......... Yes

Immediate RTC.. Yes Wait on RTC.... No

ElemChar....... 0 MaxElemNum..... 255

List count..... 1023 Lock Entries... 1024

Entries........ 169671 Elements....... 1016824

EMCs........... 103608

SCM algorithm.. KeyPriority1

In-use SCM..... 264M In-use aug spc. 5M

Max SCM ent.... 599040 Max SCM elem... 3594240

In-use SCM ent. 154440 In-use SCM elem 926640

Pend ent ratio. 1 Pend elem ratio 6

Pend EMCSTG.... 500 SLND........... 5000

AUGMENTED SPACE

CFNAME CFLVL CFSRV MIN/MASS STRSIZE MAXSIZE SCMMAXSIZE EST MAX MIN

-------- ----- ----- --------- --------- --------- ---------- -------- -------

POLICY - - 0M 380M 380M 1G N/A N/A

CURRENT - - 40M 380M 380M 1G 80M 3M

CF4 18.00 04.04 CF not SCM-capable (CFLEVEL)

Real objects only: 355M 380M 0M 0M 0M

Real plus in-use SCM objects: 658M 658M 0M 0M 0M

Real plus est max SCM objects: 1534M 1534M 0M 0M 0M

CF2 17.00 02.13 CF not SCM-capable (CFLEVEL)

Real objects only: 352M 380M 0M 0M 0M

Real plus in-use SCM objects: 655M 655M 0M 0M 0M

Real plus est max SCM objects: 1528M 1528M 0M 0M 0M

CF1 19.00 00.15 CF not SCM-capable (no SCM configured)

SCM availability assumed: 381M 381M 1G 80M 3M

Real objects only: 377M 380M 0M 0M 0M

Real plus in-use SCM objects: 695M 695M 0M 0M 0M

Real plus est max SCM objects: 1614M 1614M 0M 0M 0M

CF3 21.00 02.16 N/A 381M 381M 1G 80M 3M

----------------------------------------------------------------------------------------

The remainder of this section will describe how to interpret and use this output.

The first part of the report, for each structure, shows the structure name as defined in the CFRM policy, the name of the CF in which the structure is currently allocated, and the structure’s currently allocated, architected CF structure attributes. If the structure is in rebuild (including duplexing rebuild), the “Allocated in CF” line will include the text “(rebuild old structure)” or “(rebuild new structure)” to identify the structure instance being analyzed.

The most fundamental structure attribute is the structure type, which is either “List structure”, “Lock structure”, or “Cache structure” (source: IXLYAMDSTRL\_Type, IXLYAMDSTRL\_ST\_LI, IXLYAMDSTRL\_LTECH). The other reported attributes differ depending on the structure type.

* List structure  
    
  The following attributes are reported on for a list structure:  
  + Key/Name – Indicates whether the list structure is using Names, Entry keys, Secondary keys, or None of the above, to access list entries. The list structure architecture supports several optional indexing/access methods for list structure entries. Names allow direct access to each list entry via a user-defined entry name for each entry. Entry keys allow a single indexed ordering to be imposed on each list in the list structure, based on a user-defined entry key for each entry. Secondary keys allow two distinct indexed orderings, a primary and secondary ordering, to be imposed on each list in the list structure, based on two user-defined keys (an entry key and a secondary key) for each entry. Source: ixlyamdstrl\_st\_ni, ixlyamdstrl\_st\_ki, and Ixlyamdstrl\_st\_SKI.
  + Adjunct – Indicates whether or not the list entries have an optional 64-byte adjunct area associated with them. This area may be used by the connectors to contain “metadata” about the list entries. Source: ixlyamdstrl\_st\_ai.
  + List accounting – Indicates whether the structure is using list-limit accounting by tracking entries or data elements on lists within the structure. List limit accounting is used in conjunction with a specified list limit count, to set a maximum limit on the number of objects that can reside on a list before a “list full” condition is reported on subsequent write requests. Source: IXLYAMDSTRL\_ST\_CI
  + PLEIDs – Indicates whether or not the structure supports program-specified list entry identifiers, which provides another means to uniquely identify individual list entries within the structure. Source: IXLYAMDSTRL\_ST\_PLEIDI.
  + Immediate RTC – Indicates whether or not the “immediate RTC completion” protocol is in use for this structure instance. This protocol is supported only when the structure is system-managed duplexed between two CFs at CFLEVEL 16 or higher, and when enabled for use via the DUPLEXCF16 function setting in XCF. Source: IXLYAMDSTRL\_IRTCEI.
  + Wait on RTC – Indicates whether or not this structure instance is using a protocol for system-managed duplexing in which this structure instance will wait for the RTC signal from the other structure instance before proceeding with duplexed command execution. Source: IXLYAMDSTRL1\_WRTCLI.
  + ElemChar – The structure’s element characteristic, which determines the size of each data element in the structure. The size of a data element in bytes is 256\*(2\*\*ELEMCHAR). Source: IXLYAMDSTRL\_LELX.
  + MaxElemNum – The maximum number of data elements that may be associated with each list entry in this structure. Source: IXLYAMDSTRL\_MDLES.
  + List count – The number of list headers on which list entries may be placed in this structure. Source: IXLYAMDSTRL\_LC.
  + Lock entries – The number of optional locks that this list structure contains to provide serialization functions within the structure. If this number is zero, this is an “unserialized list” structure; if this number is nonzero, this is a “serialized list” structure. Source: IXLYAMDSTRL\_NLE.
  + Entries – The number of list entries allocated for the structure. Source: IXLYAMDSTRL\_MLSEC.
  + Elements – The number of data elements allocated for the structure. Source: IXLYAMDSTRL\_MLSELC.
  + EMCs – The number of event monitor controls (EMC) objects allocated for the structure. The EMCs are used for specialized monitoring functions within the structure, including sublist monitoring and key-range monitoring. This count is only reported if the number of EMCs is nonzero, indicating that the connectors may be using these EMC-based monitoring functions. Source: IXLYAMDSTRL\_MAXEMCCNT.
  + SCM algorithm – The algorithm used by the CF in determining which structure objects should be migrated from CF real storage to SCM and back, and when that migration should take place. The SCM algorithm corresponds to the CFRM policy SCMALGORITHM keyword. This value is reported only if the allocated structure instance is SCM-capable. Source: IXLYAMDSSCM\_SCMAT.
  + In-use SCM – The amount of SCM currently in use to contain structure objects. This value is reported only if the allocated structure instance is SCM-capable. Source: IXLYAMDSSCM\_IUSCM.
  + In-use aug spc – The amount of augmented space currently in use to support the CF’s use of SCM. This value is reported only if the allocated structure instance is SCM-capable. Source: IXLYAMDSSCM\_IUAUS.
  + Max SCM ent – The estimated maximum number of list entries that can overflow to SCM, based on the SCM maximum size, the entry-to-element ratio, and other factors. This value is reported only if the allocated structure instance is SCM-capable. Source: IXLYAMDSSCM\_EMSEC.
  + Max SCM elem – The estimated maximum number of data elements that can overflow to SCM, based on the SCM maximum size, the entry-to-element ratio, and other factors. This value is reported only if the allocated structure instance is SCM-capable. Source: IXLYAMDSSCM\_EMSELC.
  + In-use SCM ent – The number of list entries currently residing in SCM. This value is reported only if the allocated structure instance is SCM-capable. Source: IXLYAMDSSCM\_SLSEC.
  + In-use SCM elem – The number of data elements currently residing in SCM. This value is reported only if the allocated structure instance is SCM-capable. Source: IXLYAMDSSCM\_SLSELC.
  + Pend ent ratio – The pending entry portion of the pending entry-to-element ratio for this structure instance. Together with the pending element portion of the ratio, this shows the entry-to-element ratio that was requested for the structure, either at structure allocation time, or on a subsequent alter request. Source: IXLYAMDSTRL\_PETELR\_ENTRY.
  + Pend elem ratio – The pending element portion of the pending entry-to-element ratio for this structure instance. Together with the pending entry portion of the ratio, this shows the entry-to-element ratio that was requested for the structure, either at structure allocation time, or on a subsequent alter request. Source: IXLYAMDSTRL\_PETELR\_ELEMENT.
  + Pend EMCSTG – The pending EMC storage percentage value, expressed in hundredths of a percent, which shows the percentage of structure storage that was requested for the structure, either at structure allocation time or on a subsequent alter request. Source: IXLYAMDSTRL\_TEMCSTGPCT.
  + SLND – The sublist notification delay value for the structure, expressed in microseconds. For structures allocated in CFLEVEL 16 or higher CFs, this controls the delay time between delivering an initial notification to a selected user who has a monitoring interest in the sublist, and delivering notifications to all other users who have a monitoring interest in the sublist (if the user who received the initial notification does not fully process the sublist within that amount of time). Source: IXLYAMDSTRL1\_SLND.
* Lock structure  
    
  The following attributes may be reported on for a lock structure:  
  + Immediate RTC – Indicates whether or not the “immediate RTC completion” protocol is in use for this structure instance. This protocol is supported only when the structure is system-managed duplexed between two CFs at CFLEVEL 16 or higher, and when enabled for use via the DUPLEXCF16 function setting in XCF. Source: IXLYAMDSTRL\_IRTCEI.
  + Wait on RTC – Indicates whether or not this structure instance is using a protocol for system-managed duplexing in which this structure instance will wait for the RTC signal from the other structure instance before proceeding with duplexed command execution. Source: IXLYAMDSTRL1\_WRTCLI.
  + RData entries – The number of record data entries (or simply, entries) allocated for the structure. Source: IXLYAMDSTRL\_MLSEC.
  + NumUsers – The number of users (connectors) that the structure is formatted to support for lock structure usage. This determines both the number of list headers that the structure is using for storing the connectors’ record data entries, and the “width” of each lock table entry (NumUsers up to 7 will result in 2-byte lock table entries, up to 23 will result in 4-byte lock table entries, and so on). Source: IXLYAMDSTRL\_LTECH.
  + Lock entries – The number of lock table entries allocated for the structure. Source: IXLYAMDSTRL\_NLE.
  + List count – The number of list headers on which record data entries (or simply, entries) may be placed in this structure. Source: IXLYAMDSTRL\_LC.
  + Duplex cntls – The total number of duplex controls allocated to support asynchronous duplexing. Displayed as “N/A” if the structure is not allocated with duplex controls. Source: Computed from IXLYAMDSTRL1\_PCQCH.
  + DUPLEX modes – The duplexing modes specified or defaulted by the CFRM policy DUPLEX keyword for the structure. Source: QUASTRDUPLEXMODES.
* Cache structure  
    
  The following attributes may be reported on for a cache structure:  
  + Name cls mask – Shows the “name class mask” value that is in effect for the structure, in hexadecimal. A mask value other than 0000 or FFFF indicates that name classes are in use. The optional use of name classes allows the directory entries in the cache structure to be organized/grouped into special name classes based on a subset of the entry name (the actual “subsetting” is designated by the bytes of the entry name corresponding to the B’1’ bits in the name class mask), where they can be more efficiently manipulated by certain CF commands based on this naming convention. Source: IXLYAMDSTRC\_NCM.
  + Adjunct – Indicates whether or not the cache directory entries have an optional 64-byte adjunct area associated with them. This area may be used by the connectors to contain “metadata” about the cache entries. Source: ixlyamdstrc\_aai.
  + UDF order – Indicates whether or not the structure has been allocated with the “user data field order queue” function. This allows the entries in the castout classes for the cache structure to be ordered based on a special user data field specified by the connectors, for improved castout processing efficiency. Source: ixlyamdstrc\_udfoqi.
  + Immediate RTC – Indicates whether or not the “immediate RTC completion” protocol is in use for this structure instance. This protocol is supported only when the structure is system-managed duplexed between two CFs at CFLEVEL 16 or higher, and when enabled for use via the DUPLEXCF16 function setting in XCF. Source: IXLYAMDSTRC\_IRTCEI.
  + ElemChar – The structure’s element characteristic, which determines the size of each data element in the structure. The size of a data element in bytes is 256\*(2\*\*ELEMCHAR). Source: IXLYAMDSTRC\_DAEX.
  + MaxElemNum – The maximum number of data elements that may be associated with each cache directory entry in this structure. Source: IXLYAMDSTRC\_MDAS.
  + COClasses - The number of castout classes which have been allocated in the structure to contain queues of entries with “changed” data associated with them. The castout class queues are used to efficiently cast out this changed data to permanent storage when desired. Source: IXLYAMDSTRC\_MCCV.
  + StgClasses – The number of storage classes which have been allocated in the structure to contain queues of entries maintained in least-recently-used order for cache reclaiming purposes. Source: IXLYAMDSTRC\_MSCV.
  + Entries – The number of cache directory entries allocated for the structure. Source: IXLYAMDSTRC\_TDEC.
  + Elements – The number of data elements allocated for the structure. Source: IXLYAMDSTRC\_TDAEC.
  + Pend ent ratio – The pending entry portion of the pending entry-to-element ratio (also called the directory-to-data ratio) for this structure instance. Together with the pending element portion of the ratio, this shows the entry-to-element ratio that was requested for the structure, either at structure allocation time or on a subsequent alter request. Source: IXLYAMDSTRC\_PDTDR\_DIR.
  + Pend elem ratio – The pending element portion of the pending entry-to-element ratio (also called the directory-to-data ratio) for this structure instance. Together with the pending entry portion of the ratio, this shows the entry-to-element ratio that was requested for the structure, either at structure allocation time or on a subsequent alter request. Source: IXLYAMDSTRC\_PDTDR\_DATA.
  + Free dir ent – The current count of free (unused) directory entries in the structure at the time of the report. If this value is always large for the structure, then the structure may be oversized or have a skewed directory-to-data ratio. Source: IXLYAMDSTRC1\_FDEC.
  + Free data elem – The current count of free (unused) data elements in the structure at the time of the report. If this value is always large for the structure, then the structure may be oversized or have a skewed directory-to-data ratio. Source: IXLYAMDSTRC1\_FDAEC.
  + GCUDRI – The current global count of unchanged directory entries with registered interest in the structure at the time of the report. This count indicates entries that are unchanged (and therefore, reclaimable) which, if reclaimed, will cause local cache entries to be cross-invalidated. This may be relevant in helping to explain high reclaim rates. Source: IXLYAMDSTRC1\_GCUDRI.

Having thus determined and reported on the current allocation attributes for a particular allocated structure instance, the second part of the report for each structure shows the applicable policy/actual/calculated structure sizing information. This is displayed in a tabular format, with size values in units of K, M, G, or T bytes as indicated. The columns of the table are as follows:

* CFNAME – The name of the CF for which sizing information is being presented. Source: Ixlyamdcf\_CFname.   
    
  The first row (with a pseudo-CFNAME of POLICY) contains the structure sizing information from the CFRM policy, rather than from any allocated or calculated structure instance.   
    
  The second row (with a pseudo-CFNAME of CURRENT) shows the *actual allocation sizing information* for the structure as it is currently allocated, in the CF where it is currently allocated. The name of the CF in which the structure is allocated is displayed above the attribute section, immediately below the structure name, along with text indicating whether this is a rebuild old or rebuild new structure instance, if applicable.  
    
  The third through nth rows show the *calculated sizing information* for the structure, as if it were allocated with structure attributes identical to the current instance (as shown in the first part of the report), in each accessible CF in the configuration.   
    
  The reported information in each of these rows is described in more detail below, for each of the various column headings.  
    
  Note that this list of CFs will also include the CF where the structure currently resides, and thus with the SIZER output in hand, one can compare the *actual* allocation sizing information of the structure in its current CF (as shown in the CURRENT row) to the *calculated* sizing information for that structure in the very same CF. In general, these actual and calculated values will be *close,* but you should be aware that it is not at all uncommon for them to differ somewhat, or perhaps even to differ very substantially, from one another. The main reason for these actual-vs-calculated discrepancies has to do with differences in the structure allocation history: the CURRENT structure may have been rebuilt, altered, etc., since it was initially allocated, whereas the calculated structure sizes show the sizing information for a hypothetical *new*ly-allocated structure in the indicated CF. This difference in “structure history” may result in different allocations of CF-internal control areas within the structure, which may in turn result in the different sizing results visible in the SIZER output.
* CFLVL – Coupling Facility CFLEVEL  
    
  The CFLEVEL information is not shown for the POLICY and CURRENT rows.   
    
  The third through nth rows show the actual CFLEVEL of the indicated CF.
* CFSRV – Coupling Facility Service Level  
    
  The CF Service Level information is not shown for the POLICY and CURRENT rows.   
    
  The third through nth rows show the actual CF Service Level of the indicated CF.
* MIN/MASS – The minimum structure size, or minimum apportionable structure size (MASS).  
    
  The POLICY row shows the MINSIZE as defined or defaulted to by the CFRM policy. Source: QuaStrMinsize.  
    
  The CURRENT row shows the *actual* minimum apportionable structure size (MASS) which is defined as the minimum size at which the structure must be allocated in order to achieve a good approximation of the requested entry-to-element ratio. Source: IXLYAMDSTRL\_MINSS or IXLYAMDSTRC\_MINSS.  
    
  The third through nth rows show simply “N/A” for the *calculated* minimum structure size, since this is not meaningful in the context of structure sizing activities.
* STRSIZE – The allocation size. *This is generally the most important column to pay attention to and make use of.*  
    
  The POLICY row shows the INITSIZE specification from the CFRM policy. Source: QuaStrInitSize.  
    
  The CURRENT row shows the *actual allocated size* of the current structure instance. Source: IXLYAMDSTRL\_SS or IXLYAMDSTRC\_SS.  
    
  The third through nth rows show the *calculated* *size* of the structure if it were to be allocated in the indicated CF, with identical attributes. Generally speaking, this is the most useful part of the output – it indicates what you would want to define in the CFRM policy for the initial allocation size of the structure (INITSIZE), if the structure were to be relocated into the indicated CF, so as to keep its structure attributes the same as the currently-allocated structure. Source: Cspa\_Strsize.  
    
  A word of warning: if the currently-allocated structure is poorly sized, so that it does not really provide the number of structure objects of some type that the structure’s connectors require, then moving the structure to another CF using these calculated sizes will also result in a structure that is poorly sized – because its current (inadequate) structure attributes will be preserved by doing this!
* MAXSIZE – The maximum structure size.  
    
  The POLICY row shows the SIZE specification from the CFRM policy. Source: QuaStrSize.  
    
  The CURRENT row shows the *actual maximum size* of the current structure instance, which is the largest size to which it could ever be altered. Source: IXLYAMDSTRL\_MSS or IXLYAMDSTRC\_MSS.  
    
  The third through nth rows show the *calculated maximum size* of the structure if it were to be allocated in the indicated CF. This value is not usually very interesting, as it is normally just the higher of the POLICY MAXSIZE for the structure, or the calculated STRSIZE for the structure in the indicated CF. Source: Cspa\_Maxsize.
* SCMMAXSIZE – The maximum amount of SCM the structure is permitted to use.  
    
  The POLICY row shows the SCMMAXSIZE specification from the CFRM policy. Source: QUASTRSCMMAXSIZE.  
    
  The CURRENT row shows the actual maximum amount of SCM the allocated structure instance is capable of using. This may be less than the amount specified by the policy depending on conditions at the time of structure allocation. Source: IXLYAMDSSCM\_MXSCM.  
    
  The third through nth rows show the maximum amount of SCM used in calculating the structure size. It will be one of the following:  
  + The value associated with the currently-allocated structure instance. Source: IXLYAMDSSCM\_MXSCM.
  + The value necessary to preserve the estimated maximum SCM entry and element counts associated with the currently-allocated structure instance. Source: CSPA\_SCMMAXSIZE.
  + The policy-specified value of SCMMAXSIZE. Source: QUASTRSCMMAXSIZE.

Note that the SIZER utility does not consider the amount of SCM actually configured to the CF for which the analysis is being performed, except as discussed below in the section entitled “SCM Calculations”.

* EST MAX AUGMENTED SPACE – The estimated maximum amount of augmented space that would be required to support the maximum amount of usable SCM from the SCMMAXSIZE column.  
    
  The POLICY row displays “N/A” for the estimated maximum augmented space, as this quantity is not specified in the policy.  
    
  The CURRENT row displays the value associated with the currently-allocated structure. Source: IXLYAMDSSCM\_EMXAUS.  
    
  The third through nth rows display the calculated estimate of the maximum amount of augmented space that would be required to support the amount of SCM used in the size calculation. Source: CSPA\_ESTMAXAUGSPACE.
* MIN AUGMENTED SPACE – The minimum (fixed) amount of augmented space that would be associated with a structure with the specified amount of SCM.  
    
  The POLICY row displays “N/A” for the minimum augmented space, as this quantity is not specified in the policy.  
    
  The CURRENT row displays the value associated with the currently-allocated structure. Source: IXLYAMDSSCM\_FXAUS.  
    
  The third through nth rows display the calculated amount of fixed augmented space that would be required to support the amount of SCM used in the size calculation. Source: CSPA\_FIXEDAUGSPACE.

**SCM Calculations**

When the structure being analyzed is SCM-capable or is defined as SCM-eligible with a non-zero SCMMAXSIZE value in the CFRM policy, and the CF for which the analysis is being performed is SCM-capable, SIZER performs a single calculation and displays the results in a single line, as it would when SCM is not a factor.

When the structure being analyzed is SCM-capable or is defined as SCM-eligible, but the CF for which the analysis is being performed is not SCM-capable, SIZER provides calculated structure sizes for several possible sizing options. These are illustrated in the sample report above for the MQGPBOOK structure.  
  
A CF might not be able to use SCM either because its CFLEVEL is too low or because it doesn't have SCM configured. When the structure would be expected to use SCM but the CF doesn’t support it, the report line for that structure / CF combination will contain the text   
  
CF not SCM-capable (CFLEVEL | no SCM configured)

after the CFLEVEL and release information. For a CF that isn't SCM-capable, SIZER performs up to four “what-if” calculations, the results of which follow the “CF not SCM-capable” line.

* SCM availability assumed - Preserve the real and flash capacity of the allocated structure by assuming the required SCM is available in the target CF.  
    
  This calculation determines the structure size that would be required to preserve existing structure capacity if the installation were to configure the target CF with the necessary SCM. There are two situations of interest:
* The allocated structure is SCM-capable.  
    
  In this case, the calculation seeks to preserve the existing capacity and uses the maximum SCM value associated with the allocated structure.
* The allocated structure isn't SCM-capable but the policy specifies that it is SCM-eligible.  
    
  In this case, the calculation uses the policy SCMMAXSIZE specification.

SIZER performs this calculation when the target CF is at or above CFLEVEL 19 and therefore can perform SCM-related calculations, but has no SCM configured.

* Real objects only - Preserve the real-storage capacity of the allocated structure, in CF real storage.  
    
  If the allocated structure were to be allocated in a non-SCM-capable CF, all of the structure’s existing capacity would have to be provided by CF real storage. This calculation assumes that only the allocated structure’s real storage capacity is to be preserved. It determines the structure size that would be required to contain the maximum number of entries and elements that the allocated structure can contain without overflowing to SCM.  
    
  SIZER always performs this calculation in SCM what-if situations.
* Real plus in-use SCM objects - Preserve the real-storage capacity plus the in-use SCM capacity of the allocated structure, in CF real storage.  
     
  This calculation determines the structure size that would be required to contain not only the maximum number of entries and elements that the allocated structure can contain without overflowing to SCM, but also the objects currently residing in SCM in the allocated structure.  
    
  SIZER performs this calculation when the allocated structure is SCM-capable and is currently using SCM to store structure objects.
* Real plus est max SCM objects - Preserve the real-storage capacity plus the estimated maximum SCM capacity of the allocated structure, in CF real storage.   
    
  This calculation determines the structure size that would be required to preserve the real capacity of the allocated structure, and in addition provide capacity in CF real storage for the entire overflow capacity represented by the allocated structure’s SCM. This calculation is likely to result in a very large structure size.  
    
  SIZER performs this calculation when the allocated structure is SCM-capable.

**Asynchronous Duplexing Calculations**

When the structure being analyzed is capable of asynchronous duplexing (i.e., allocated with duplex controls), the CFRM policy DUPLEX specification for that structure defines it as eligible for asynchronous duplexing, and the CF for which the analysis is being performed is capable of supporting asynchronous duplexing, SIZER performs a single calculation and displays the results in a single line, as it would when asynchronous duplexing is not a factor.

When either (1) the structure being analyzed is capable of asynchronous duplexing but the CFRM policy DUPLEX specification for that structure does not define it as eligible for asynchronous duplexing, or (2) the structure is not capable of asynchronous duplexing but the CFRM policy DUPLEX specification defines it as eligible for asynchronous duplexing, SIZER provides calculated structure sizes for two possible options.

* With duplex controls – Assumes that the structure is intended to support asynchronous duplexing.  
    
  This calculation determines the structure size that would be required to preserve the capacity of the structure and provide the number of duplex controls supported by the CF for which the calculation is being performed.
* Without duplex controls – Assumes that the structure is not intended to support asynchronous duplexing.  
    
  This calculation determines the structure size that would be required to preserve the capacity of the structure without duplex controls.

When the structure being analyzed is capable of asynchronous duplexing or the CFRM policy DUPLEX specification for that structure defines it as eligible for asynchronous duplexing, but the CF for which the analysis is being performed is at a CFLEVEL / service level combination that is not capable of supporting asynchronous duplexing, the report line for that structure / CF combination will contain the text   
  
CF not ASYNCDUPLEX-capable

after the CFLEVEL and release information. SIZER calculates the structure size without duplex controls as described above. This situation is illustrated in the sample report above for the DSNDBSG\_LOCK1 structure.